RO888-0269 BAF
S. R. Ritland

IBM Technical Disclosure Bulletin Vol. 32 No. 9B February 1990

90A060623

CALL-TYPE API TO SQL/DS WITH EXTERNALLY DESCRIBED OPERATIONS

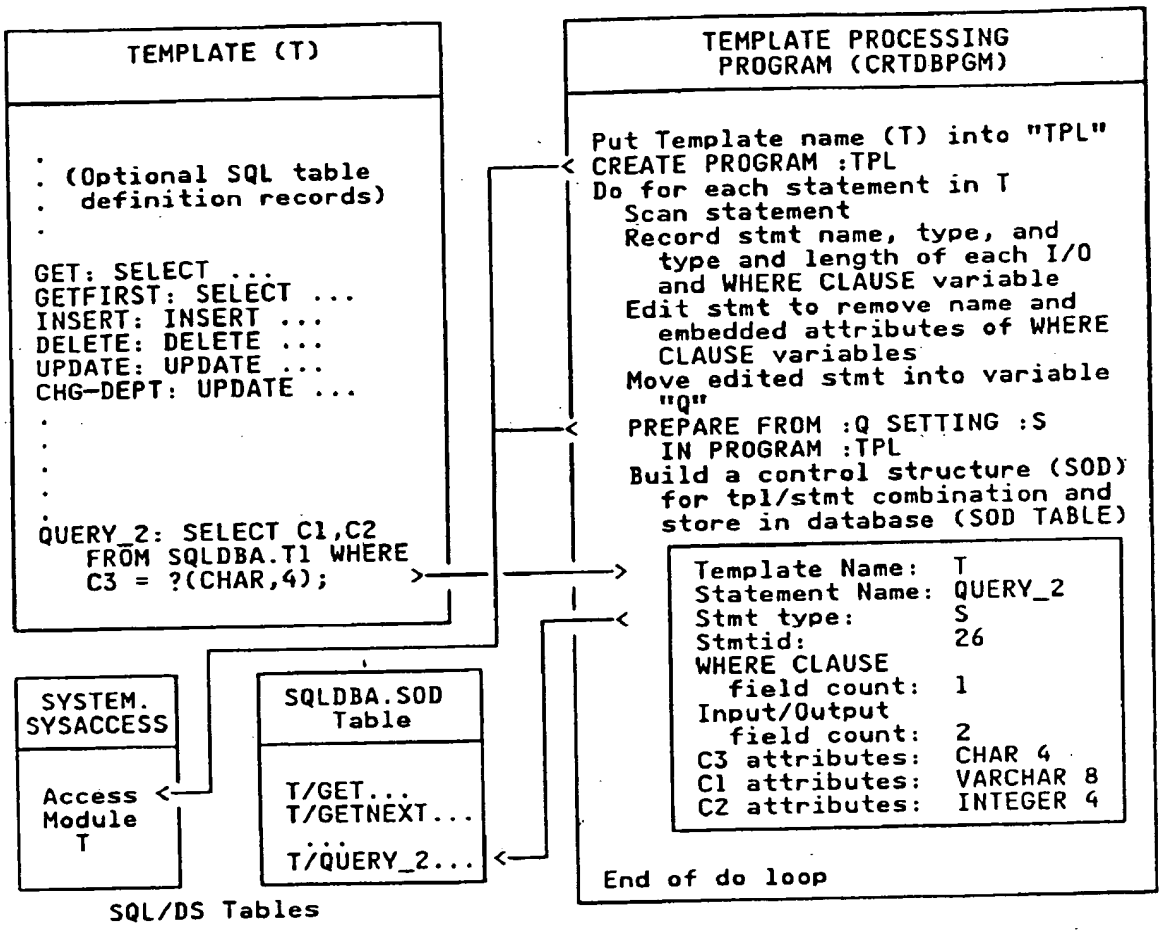

Fig. 1

A method is described which specifies and names structured-query-language data-manipulation-language (SQL DML) operations independently from any of the programs that use them, a generalized SQL statement processor for generating access modules and other data structures used at run-time, and the run-time support itself.

This method comprises:

CALL-TYPE API TO SQL/DS WITH EXTERNALLY DESCRIBED OPERATIONS - Continued

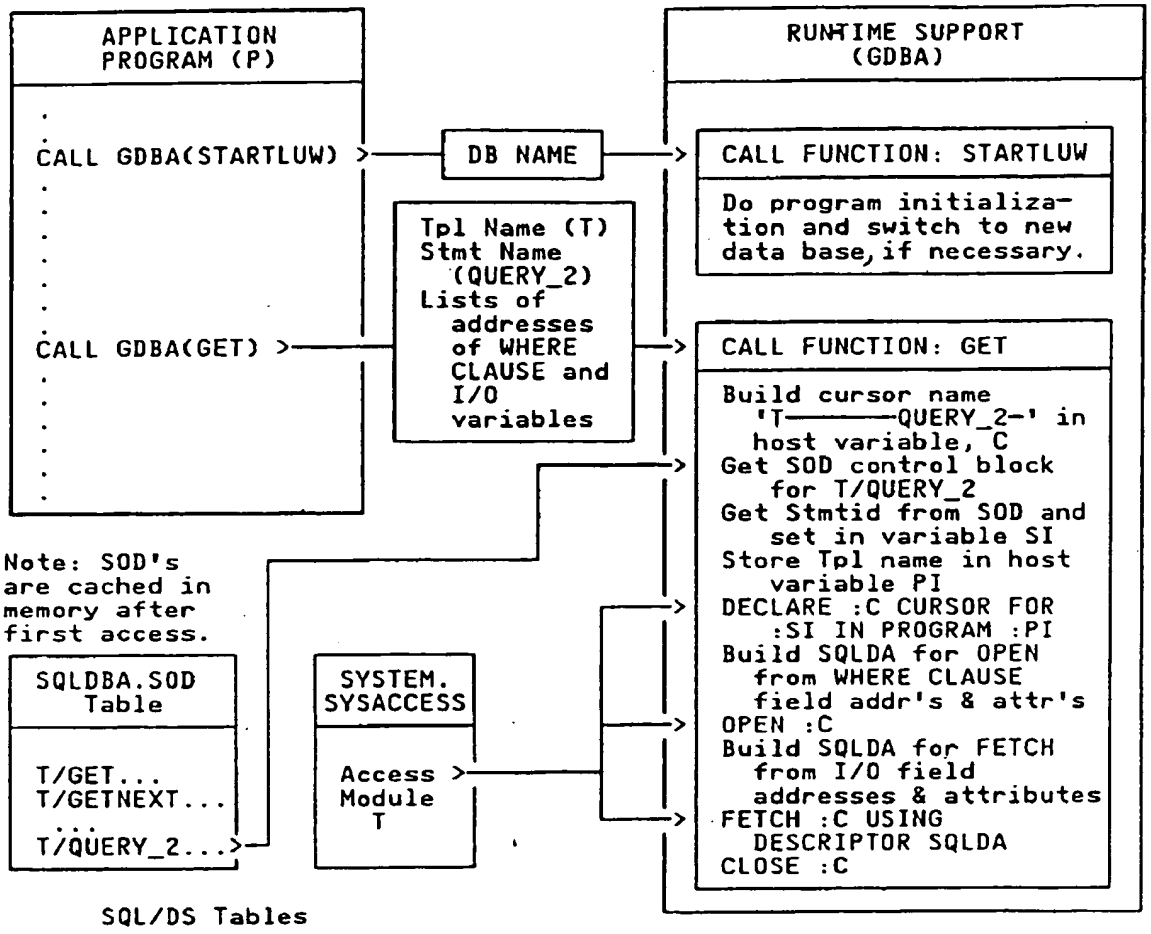

Fig. 2

- A method of specifying and processing SQL DML statements in a form external to the application programs that use them. A set of SQL statements is stored in a CMS file called a 'template'. An SQL/DS table description is optionally placed in a template file, in which case a set of basic SQL statements to access data in that table are automatically generated and appended to the template file.

- A data structure called an SQL Operation Descriptor (SOD) produced by the processing of a template. The SQL SOD contains information required at run-time.

- Two programs for processing the template files.

- An application programming interface (API) consisting of control functions and a set of data access functions referencing the SQL statements predefined in template files.

CALL-TYPE API TO SQL/DS WITH EXTERNALLY DESCRIBED OPERATIONS - Continued

- A run-time module which uses the SODs and parameters passed on calls to perform the data control and access functions.

- The environment and a program to initialize it.

Template files contain one or multiple SQL statements, each of which is referenced by a name. A special type of template is used as input to a table-creation process. These are called 'table templates' and contain, possibly in addition to SQL DML statements, the description of an SQL table to be created. Besides creating the SQL table, the table template processor also generates SQL statements which are appended to the template to do basic operations on the associated table.

Each SQL statement in a template is prefixed by a name delimited by a ':'. The SQL statements are coded in a form very similar to that of the extended dynamic form of SQL/DS DML, in which ?'s are used as place holders for host program input variables.

In a table template, a table description is placed at the beginning of the file, just following the header record which indicates the template name and type. Each table column (field) is represented by a row in the table definition section. That row specifies the column name, its data type, its length, whether it is a part of the table's unique index, and whether it can have a NULL value.

THE SQL SOD STRUCTURE. The SOD data structure is used at run-time along with information stored in an SQL/DS access module to inform a data base management system (DBMS) how to carry out a particular operation.

Information about the operands participating in any SQL DML operation is passed to the DBMS in a data structure called SQLDA. The SOD and process for generating it provide certain required data at run-time.

SODs as well as the associated access module are created by one of the template processing programs. The SOD contains:

- An operation-type code for validating that the SOD is used with the appropriate run-time function.
- The associated statement number in the access module.
- A set of bit flags.
- The number of WHERE clause data items in the operation.
- The number of input or output data items in the operation.
- An array specifying the type and length of each operand.

TEMPLATE PROCESSORS. Two programs for the processing of template files are CRTDBTBL and CRTDBPGM. The access module is identified by the name of the associated template, and the SODs are identified by the combination of template name and statement name.

CRTDBTBL creates a table in the SQL/DS data base based on the table description in the input file. It also creates a unique index for the table based on the table columns which are flagged as 'key' fields. Using its knowledge of the table columns, it constructs a set of 'default' SQL statements to do the most basic operations that the programmer might want to do on the table.

The other template processor, CRTDBPGM, produces two outputs. One output is an SQL/DS access module containing the access strategies for the processed template which SQL/DS uses at run-time. The other is a set of SODs, which are stored in a data base table.

ENVIRONMENT INITIALIZATION PROGRAM. SQL/DS uses several different data areas in its interaction with application programs. Rather than re-allocating these for each program invocation or SQL operation, a program is provided that initializes the required environment. It pre-allocates these areas as well as another communication area one time, and they are retained and re-used throughout the existence of this environment.

RUN-TIME SUPPORT. The run-time program accepts a variable length list of parameters which includes a function code, and, for most functions, the identification of a template and statement within that template. Where required, the addresses of host program input and output variables are also passed.

The following functions are supported by the run-time program:

- Start logical unit of work (LUW). While the concept of an LUW is an integral part of SQL/DS, normally LUWs are implicitly started. This function is provided because an initialization procedure is required for each LUW. A parameter passed on this function call specifies the name of the SQL data base desired for the LUW. This allows the application program to dynamically switch data bases when desired and provides a way for the run-time support to insure that the user or some other program has not switched data bases unbeknown to the application program. This data base checking is bypassed, if desired.
- The SQL COMMIT WORK operation.
- The SQL COMMIT WORK RELEASE operation.
- The SQL ROLLBACK WORK operation.
- The SQL ROLLBACK WORK RELEASE operation.
- Get SOD. This provides advanced application programs access to the SOD for a particular operation.
- GET a single row from the data base.
- GET FIRST row of a set of rows from the data base.
- GET NEXT row of a set of rows from the data base.
- INSERT a row into the data base.
- UPDATE selected fields in one or more rows of the data base.
- DELETE one or more rows from the data base.

Rather than exposing special indicator values to the application program for flagging null column values, nulls are encoded and decoded by the run-time program to and from special unique values such as, in the case of integers, the largest possible negative value.

PSEUDO-CODE FOR OPERATION. Figs. 1 and 2, respectively, show pseudo-code examples for the processing of a template and the execution of an application program using the outputs resulting from that processing.